REMARKS

Claims 1-50 are cancelled. New claims 51-103 are added. No new matter is added.

The Examiner rejected various cancelled claims under 35 U.S.C. § 102(e) as being anticipated by U.S. Patent No. 6,327,661 to Kocher *et al.* ("Kocher"), and other claims under 35 U.S.C. § 103 over Kocher in combination with various prior art references. Kocher discloses several methods of thwarting detection of secret information in cryptographic devices by analysis of externally observable parameters, including adding noise to reduce the SNR, clock skipping, and adding computational entropy. Abstract.

The pertinent methodology of Kocher – adding entropy to the operation order – is disclosed at col. 10, line 50 – col. 13, line 20. "Another approach ... involves <u>the introduction of entropy</u> into <u>the order of processing operations</u> or into <u>the execution path</u> while maintaining desired functionality." col. 10, lines 51-55 (emphasis added). That is, Kocher discloses reordering processing operations by a random factor, or introducing randomness into the execution path of a given processing operation. Kocher only provides an example of randomizing operation order. "As an illustrative example of operation order entropy, consider a bit permutation." col. 10, lines 65-66 (the example spans col. 10, line 65 – col. 13, line 20). In this example, Kocher randomizes the order in which bits in the input array are processed – or permutated – to generate the output array. "Bit order permutation table 'perm' randomizes the time at which any particular data bit is manipulated." col. 12, lines 29-30. Kocher does not disclose any example of introducing entropy into the execution path – in fact, the disclosed example <u>removes</u> variability in the execution path that may be analyzed for detection. "[T]he high-entropy permutation operation, above, uses a constant execution path to inhibit leakage via variations in the execution path." col. 12, lines 17-19.

The present invention does not thwart detection by introducing entropy – or randomness – into either the order of process operations or the execution path of any given operation. Rather, in various embodiments, the present invention thwarts the detection of operations

indicative of secret bit patterns by performing dummy computations and throwing away the results (claims 51, 73, 87); by altering the order of operations within cryptographic calculations without introducing any randomness (claims 63, 80, 94), by algorithmic alteration of cryptographic calculations to use data values related to, but distinct from, secret bit patterns (claims 67, 84, 97); or by combinations of the inventive techniques (e.g., claims 70, 72). Kocher – either alone or as modified by various references – does not teach or suggest any of these embodiments.

Claims 51, 73 and 87

Claims 51, 73 and 87 recite introducing dummy calculations into an operation, and throwing away (not accumulating) the results of the dummy calculations to alter one or more externally observable parameters. As described in the Specification at p. 3, lines 6-20, the fundamental cryptographic operation of exponentiating a large number by a large secret key may be performed by selectively calculating successive squares of the large number (base) if the corresponding bit of the secret key (exponent) is a one, multiplicatively accumulating the squares, and reducing the result modulo-N. Where the corresponding bit position of the key (exponent) is zero, the square is not calculated (and, obviously, not accumulated). Both the temporal duration and the power consumption of a loop iteration that checks a key bit, finds a bit value of zero, and iterates to the next bit, are less than those of a loop iteration that checks a key bit, finds a bit value of one, and in response calculates a partial square, multiplicatively accumulates it, and reduces the result modulo-N. This difference in either externally observable parameter may be exploited to ascertain the secret key bit sequence.

According to claims 51, 73 and 87 and their dependent claims, during selected loop iterations where the key bit value is a zero (indicating no operation should be performed), the successive square is calculated anyway, but is not accumulated to the results, so as to preserve the integrity of the calculation. In this manner, the duration and power consumption of the loop iteration are virtually identical to those of a loop where the bit value was a one. In one

embodiment, the selected zero-value bit positions where dummy calculations are performed are stored in a digital indicator word the same length as the secret key. The one's may be randomly distributed within the indicator word bit positions.

Kocher does not disclose introducing dummy calculations in an operation to thwart detection of a secret bit pattern. Kocher discloses only adding entropy to operation ordering – that is, randomizing the order of operations. According to the present invention of claims 51, 73 and 87, the order of operations of the exponentiation is not changed. The same sequential squares are calculated, multiplicatively accumulated, and modulo-N reduced – and in the same order – as known in the prior art. The present invention of claims 51, 73 and 87 add dummy calculations, whose results are not retained, to thwart detection. Kocher does not teach or suggest introducing dummy calculations into a computational process to thwart detection of a secret key.

Messerges discloses "randomizing" the square-and-multiply exponentiation algorithm by randomly selecting a non-MSB and non-LSB starting bit position in the secret key (exponent), proceeding left to the MSB, and then proceeding right to the LSB. This randomizes the apparent order of 1's that can be detected by analysis of externally observable parameters. Messerges does not teach or suggest inserting dummy calculations within a "normal" ordering of the secret key bits, to alter the number and order of 1's that can be detected by analysis of externally observable parameters.

Ohki discloses executing dummy calculations, along with bit mirroring and process order randomizing, to reduce the correlation between process steps and current consumption. Ohki discloses loop iterations as an example dummy process. "This dummy process may be a process of repeating a loop without a loop process as many times as the value of the random number. Since the number of loop operations changes with the random number, the start of the S box process becomes random so that the dependency of the wave shape of consumption current upon data process reduces." col. 11, lines 30-33. Ohki does not disclose executing a

calculation that a process algorithm specifically calls for not executing, and throwing away the result of the calculation.

Claims 63, 80, 94

Claims 63, 80, 94, and claims depending therefrom, exploit the fact that the calculation of successive squares, their multiplicative accumulation and modulo-N reduction are all operations that alter externally observable parameters, but the selection of which calculated square from a plurality, does not. According to this embodiment, a group of sequential squares (the group ranging from a convenient number, such as eight, to the entire bit length of the secret key) are all calculated and temporarily stored. The corresponding bits of the secret key are then consulted to determine which of the calculated squares should be multiplicatively accumulated and the results modulo-N reduced. In one embodiment, each square – whether its value is "used" or not – is then immediately recalculated using the values appropriate to the relative bit position for the next successive group. Alternatively, the group may be recalculated following the disposition of the squares from the previous group. In either case, since the successive square corresponding to each secret key bit position is calculated, no information about the bit pattern may be gleaned from externally observable parameters. While the number of multiplicative accumulations and modulo-N reductions corresponds to the number of one's in the secret key, since the selection process of which squares to accumulate takes little computational power, the relative positions of the one's within the secret key cannot be determined from externally observable parameters.

Note that no randomization is inserted into this algorithm. Rather, a sequential algorithm has been modified to a sort of "batch" or group processing, taking advantage of "the reasonable assumption that it is impossible to distinguish selection of one value to multiply the accumulator from another, only the total number of values selected, [it] is the total number of 1's contained in [the secret key] betrayed, but not their bit positions." Specification, p. 14, lines 14-16. In contrast, Kocher teaches only altering the order of operations by inserting entroy – or

randomness – into the ordering. According to claims 63, 80 and 94, successive squares are accumulated in precisely the same order as in the prior art successive algorithm; no randomness is introduced.

Lin discloses a block-cipher algorithm to protect against Differential Power Analysis, wherein the order in which blocks are combined varies for each round or iteration of the encryption operation, preferably with a random variation. The present invention does not vary the order in which successive squares are multiplicatively accumulated and modulo-N reduced.

Claims 67, 84, 97

Claims 67, 84, 97, and claims depending therefrom, thwart detection of a secret key by not using the key to directly control any operation that alters an externally observable parameter. According to this embodiment, the secret key is factored into a product of sparse binary integers plus a remainder. These integers and remainder are carefully chosen to meet specific criteria, such as having a minimum of 1's between them, and having fewer 1's than the secret key. For example a key K may be factored into K1, K2 and K3 such that K=K1*K2+K3. The exponentiation $N^{K1*K2+K3}$ thus becomes $(N^{K1})^{K2} * N^{K3}$. Each of these exponentiations may be performed according to known prior art techniques, or according to one or more of the inventive techniques described above for further security. Even if an attacker knew of the algorithm and the specific order of execution of the exponentiation of the multiplicands, he would not know, and is not likely to discover, the values K1, K2 and K3. These may be calculated once, and stored on the cryptographic computational device with (or indeed, in lieu of) the secret key.

The technique of claims 67, 84, 97 do not introduce randomness into the order of any operations (or any execution path). The embodiment substitutes carefully chosen factors for the secret key, and substitutes two exponentiations and a multiplication for the exponentiation operation. Neither injects any randomness. Kocher does not disclose or suggest altering both a secret value and the operations it controls, to thwart detection of the secret value by analysis

of externally observable parameters. No other cited art discloses factoring the secret key and performing exponentiation using the factors and remainder.

Accordingly, all claims currently pending are novel and nonobvious over the cited art. Prompt allowance of the same is hereby respectfully requested.

Respectfully submitted,

COATS & BENNETT, P.L.L.C.

Edward H. Green, III
Attorney for Applicants
Registration No.: 42,604

Dated: September 17, 2004

P.O. Box 5
Raleigh, NC 27602
Telephone: (919) 854-1844